Lab 3

Jordan Small

November 4, 2023

CDA 3203 Computer Logic Design

Spring 2023

Dr. Maria Petrie

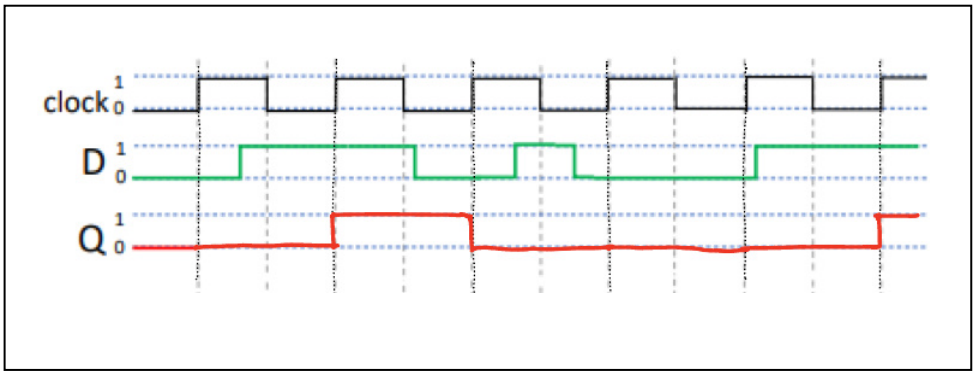
Florida Atlantic University

Handwork

1.1 - D Flip Flop

D Latch

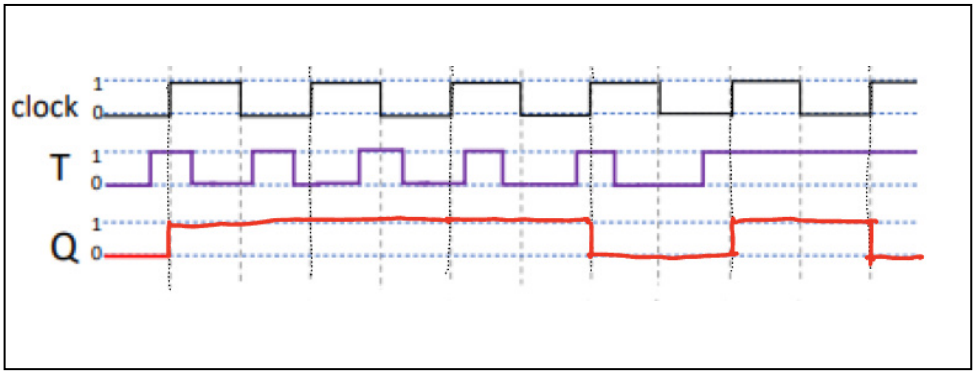
D	Q	Action	Q+
0	0	Q=0	Ø
	1		
1	0	Q=1	1
	1		



1.2 - T Flip Flop

Active-HIGH T Latch

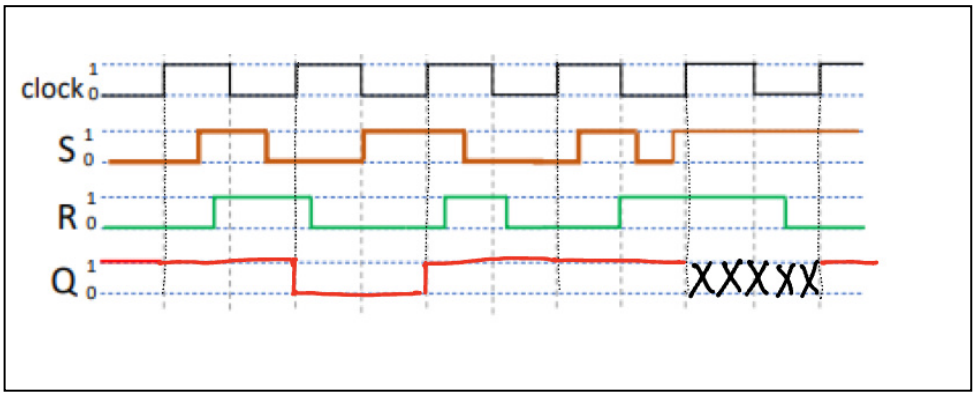
T	Q	Action	Q+
0	0	hold	0
	1		1
1	0	Toggle	1
	1		0



1.3 – S-R Flip Flop

Active-HIGH S-R Latch

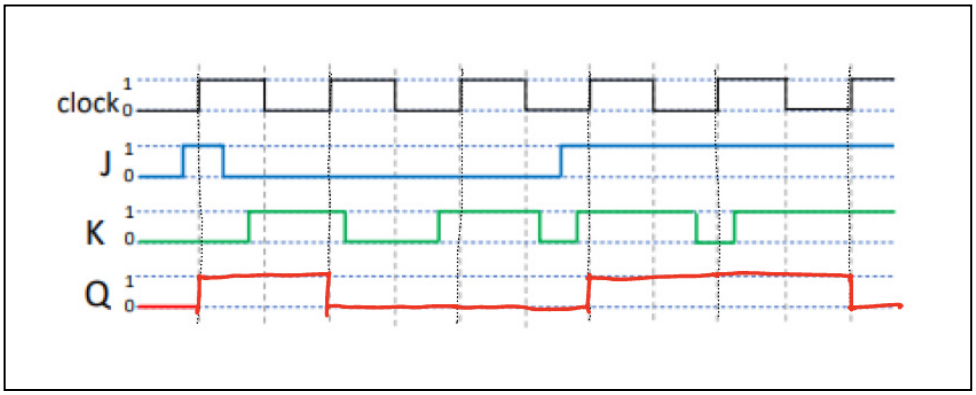
S	R	Q	Action	Q+
0	0	0	Hold	0
		1		1
0	1	0	Reset	0
		1		0
1	0	0	Set	1
		1		1
1	1	0	Invalid	?
		1		?



1.4 – J-K Flip Flop

Active-HIGH J-K Latch

J	K	Q	Action	Q+
0	0	0	HOLD	0
		1		1
0	1	0	RESET	0
		1		0
1	0	0	SET	1
		1		1
1	1	0	TOGGLE	1
		1		0



1.5 – Excitation Tables

D Excitation Table		
Q→Q⁺	Commands	D
0→0	Q=0	0
0→1	Q=1	1
1→0	Q=0	0
1→1	Q=1	1

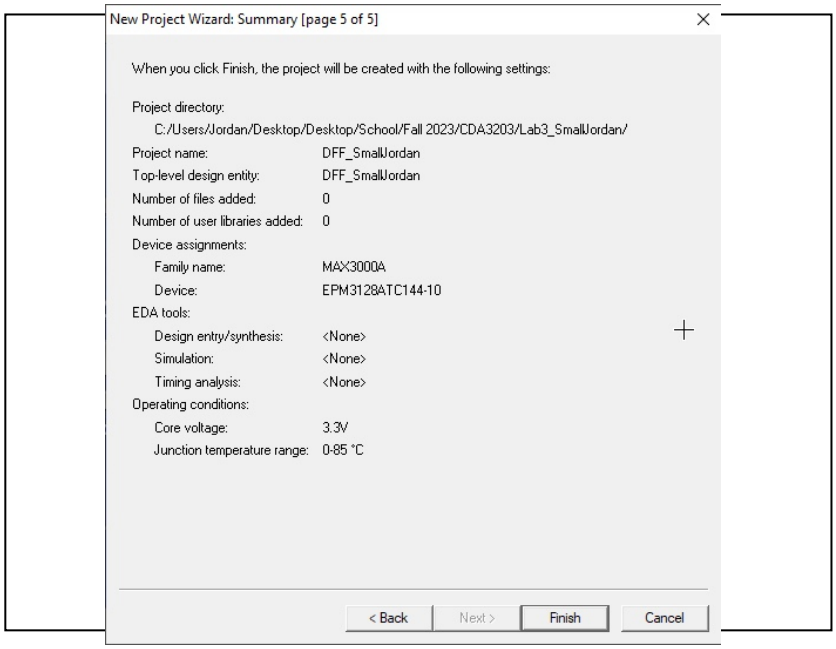
T Excitation Table		
Q→Q⁺	Commands	T
0→0	Hold	0
0→1	toggle	1
1→0	toggle	1
1→1	Hold	0

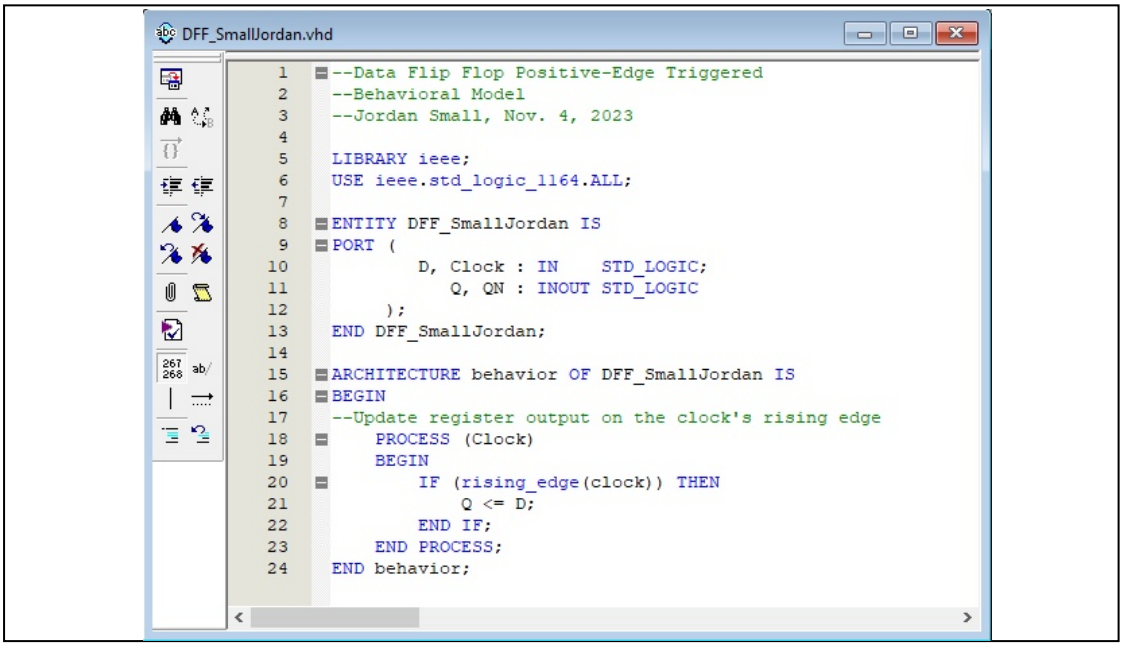
S-R Excitation Table		
Q→Q⁺	Commands	S-R
0→0	H:00 | R:01	0X
0→1	SET	10
1→0	RESET	01
1→1	H:00 | S:10	X0

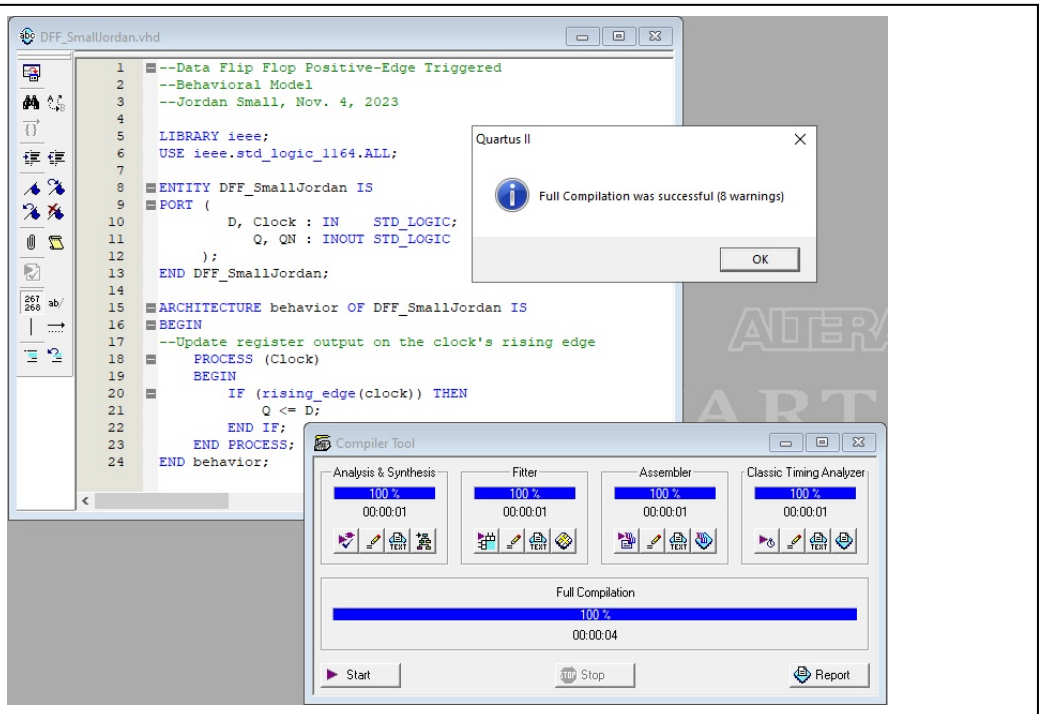
J-K Excitation Table			
Q→Q⁺	Commands		J-K
0→0	H:00	R:01	0X
0→1	S:10	T:11	1X
1→0	R:01	T:11	X1
1→1	H:00	S:10	X0

2 – Design and Simulation of Sequential Components in Altera Quartus using VHDL

2.1 – Data Flip Flop (DFF)

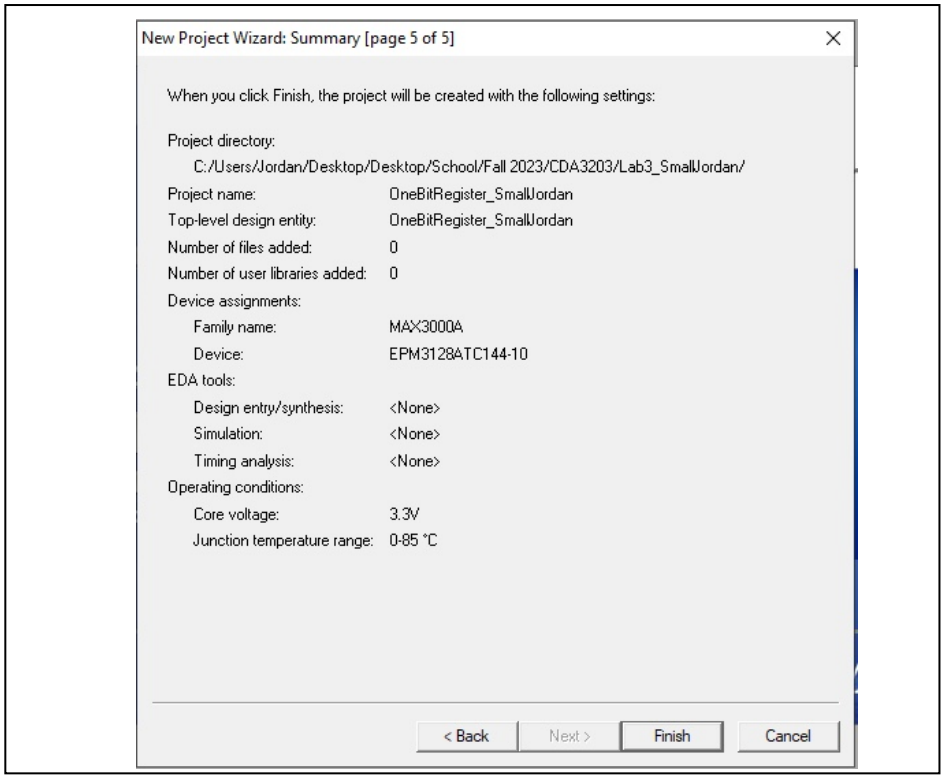


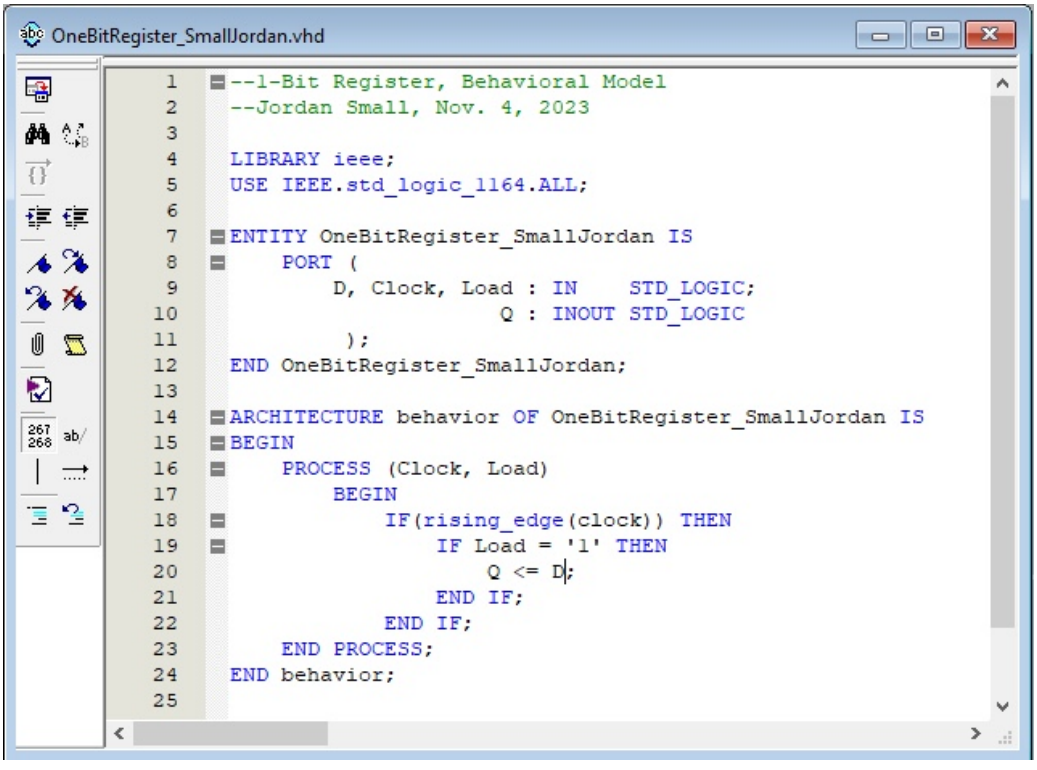


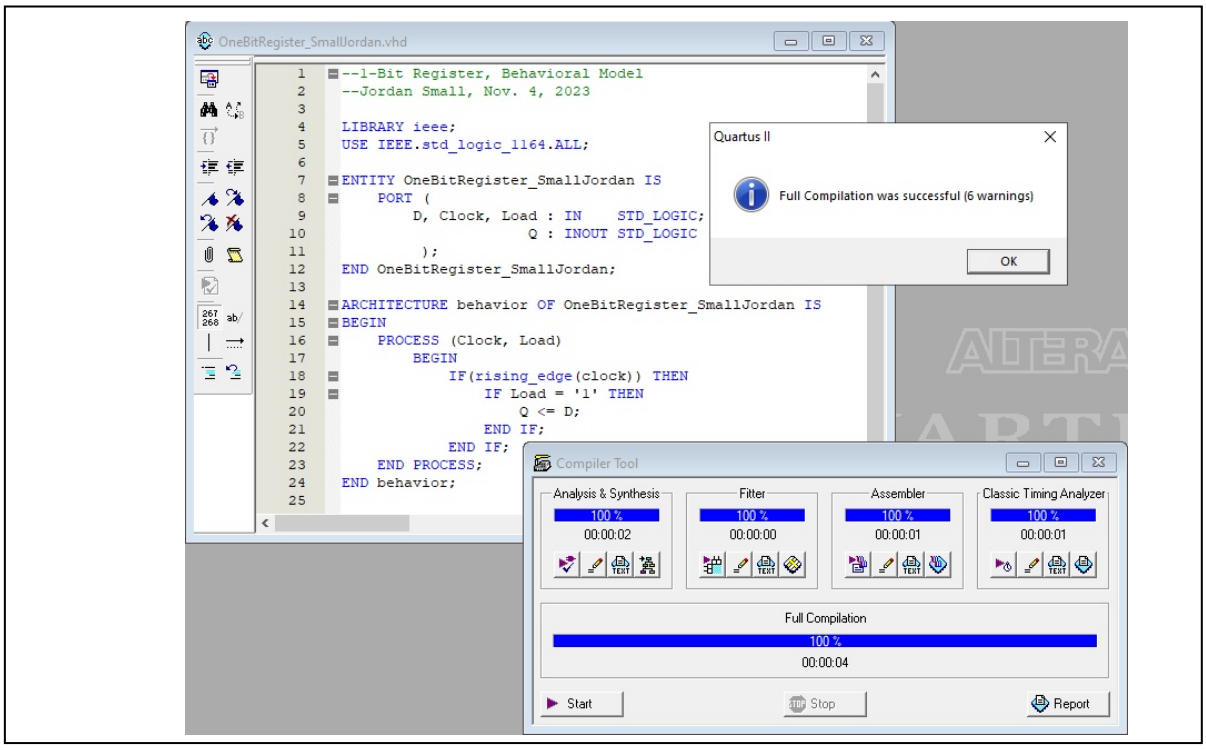

DFF_SmallJordan.vhd
--Data Flip Flop Positive-Edge Triggered
--Behavioral Model
--Jordan Small, Nov. 4, 2023
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY DFF_SmallJordan IS
PORT (
D, Clock : IN STD_LOGIC;
Q, QN : INOUT STD_LOGIC
);
END DFF_SmallJordan;
ARCHITECTURE behavior OF DFF_SmallJordan IS
BEGIN
--Update register output on the clock's rising edge
PROCESS (Clock)
BEGIN
IF (rising_edge(clock)) THEN
Q <= D;
END IF;
END PROCESS;
END behavior;
Quartus II
Full Compilation was successful (8 warnings)
OK
Compiler Tool
Analysis & Synthesis
Fitter
Assembler
Classic Timing Analyzer
100 %
00:00:01
Full Compilation
00:00:04
Start
Stop
Report

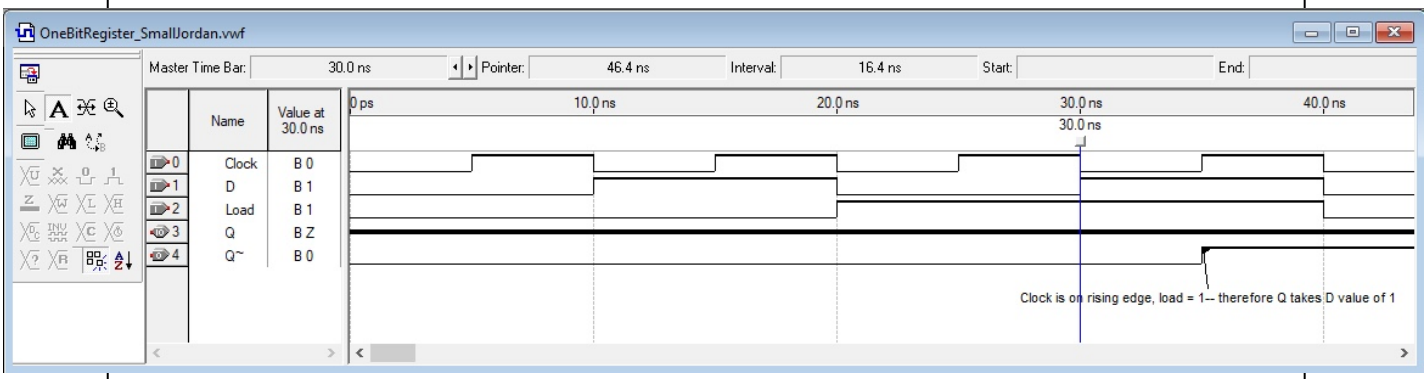

DFF_SmallJordan.vwf
Master Time Bar: 0 ps
Pointer: 0 ps
Interval: 0 ps
Start:
End:
Name
Value at 0 ps
0 Clock B 0
1 D B 0
2 Q B 0
3 QN B Z
4 Q~ B 0
5 QN~ B Z
0 ps
10.0 ns
20.0 ns
30.0 ns
40.0 ns
50.0 ns
60.0 ns

2.2 – 1-Bit Register

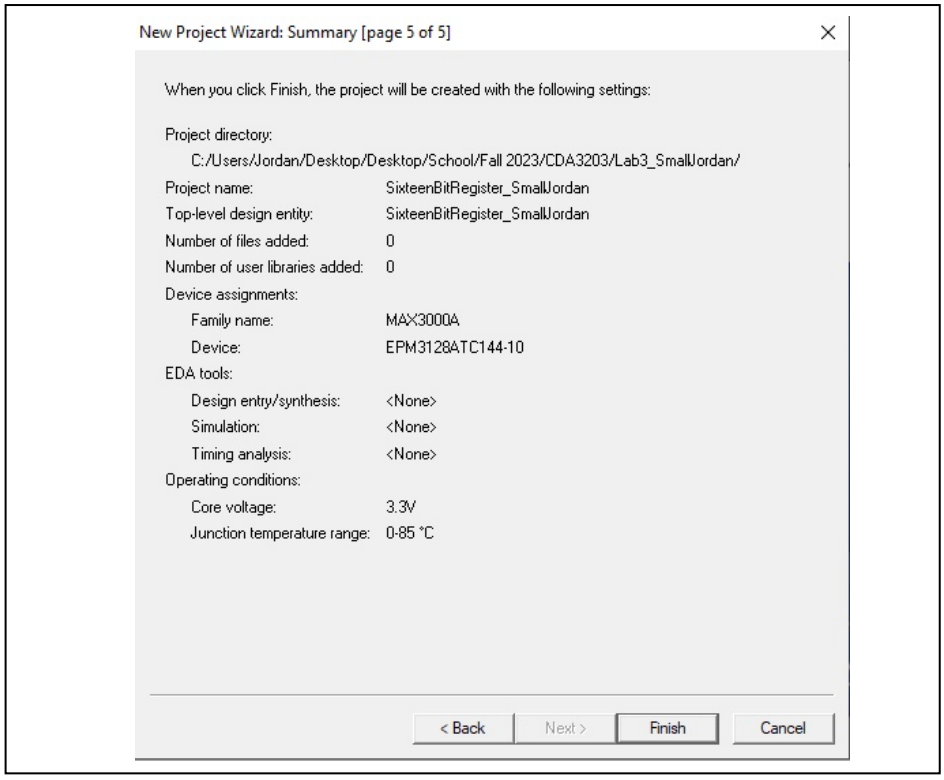


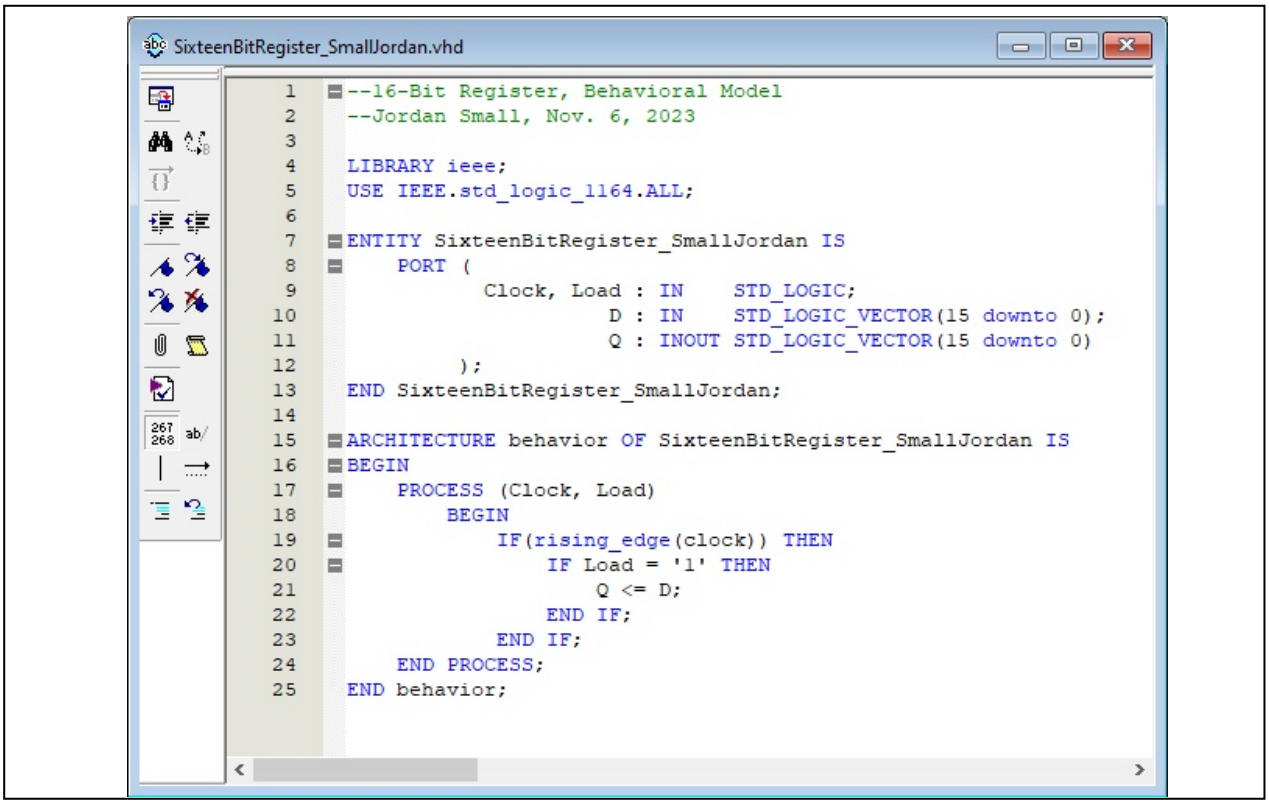


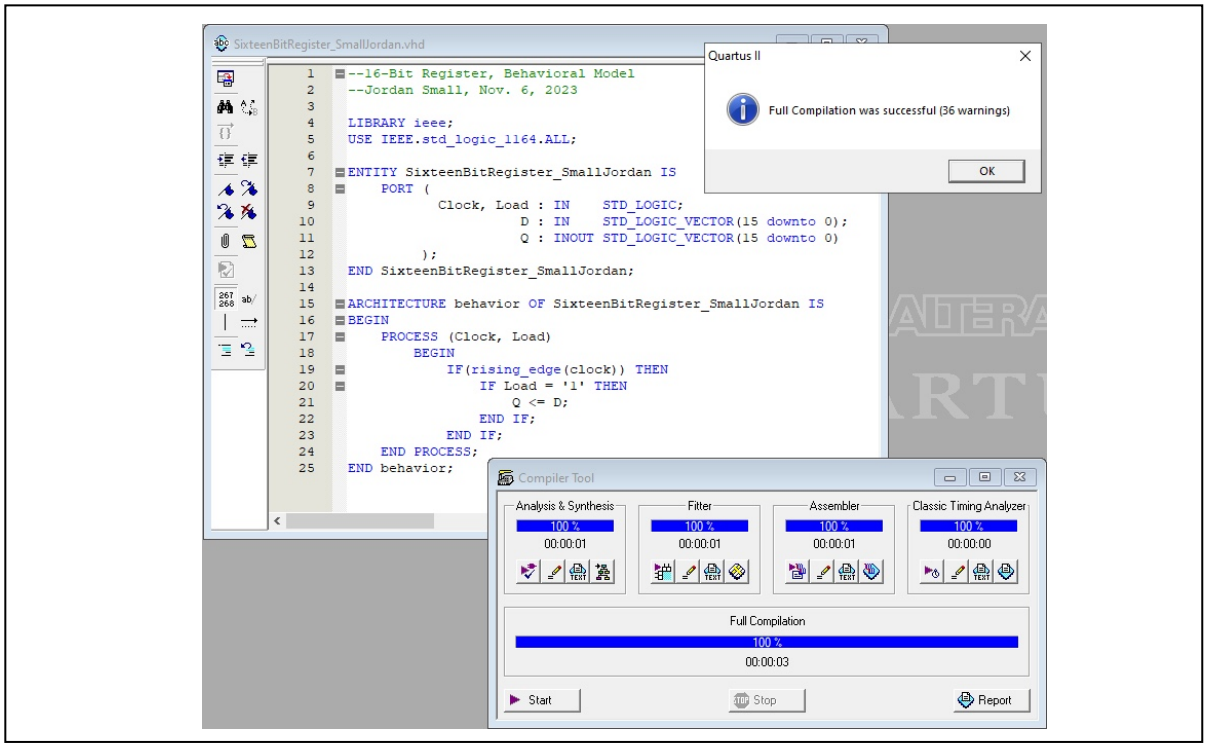

OneBitRegister_SmallJordan.vhd
--1-Bit Register, Behavioral Model
--Jordan Small, Nov. 4, 2023
LIBRARY ieee;
USE IEEE.std_logic_1164.ALL;
ENTITY OneBitRegister_SmallJordan IS
PORT (
D, Clock, Load : IN STD_LOGIC;
Q : INOUT STD_LOGIC
);
END OneBitRegister_SmallJordan;
ARCHITECTURE behavior OF OneBitRegister_SmallJordan IS
BEGIN
PROCESS (Clock, Load)
BEGIN
IF(rising_edge(clock)) THEN
IF Load = '1' THEN
Q <= D;
END IF;
END IF;
END PROCESS;
END behavior;
Quartus II
Full Compilation was successful (6 warnings)
OK
Compiler Tool
Analysis & Synthesis
100 %
00:00:02
Fitter
100 %
00:00:00
Assembler
100 %
00:00:01
Classic Timing Analyzer
100 %
00:00:01
Full Compilation
100 %
00:00:04
Start
Stop
Report

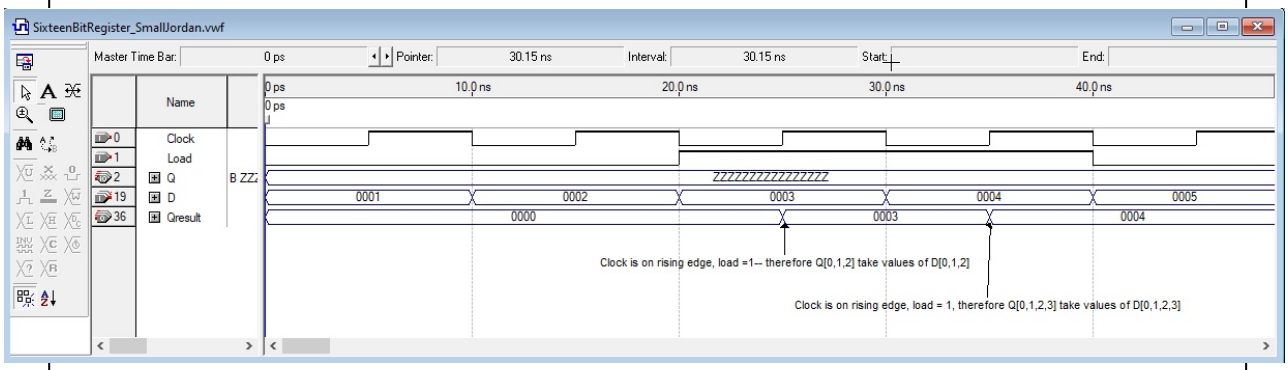

OneBitRegister_SmallJordan.vwf
Master Time Bar: 30.0 ns
Pointer: 46.4 ns
Interval: 16.4 ns
Start:
End:
Name
Value at 30.0 ns
0 Clock B 0
1 D B 1
2 Load B 1
3 Q B Z
4 Q~ B 0
0 ps
10.0 ns
20.0 ns
30.0 ns
40.0 ns
30.0 ns
Clock is on rising edge, load = 1-- therefore Q takes D value of 1

2.3 – 16-Bit Register

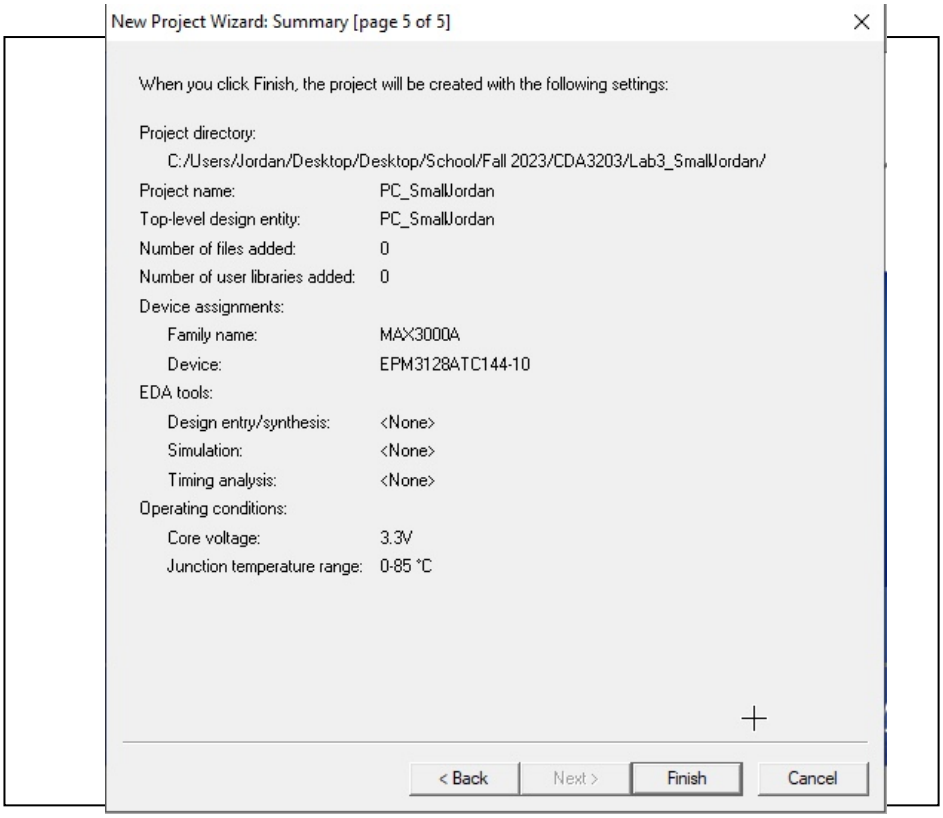


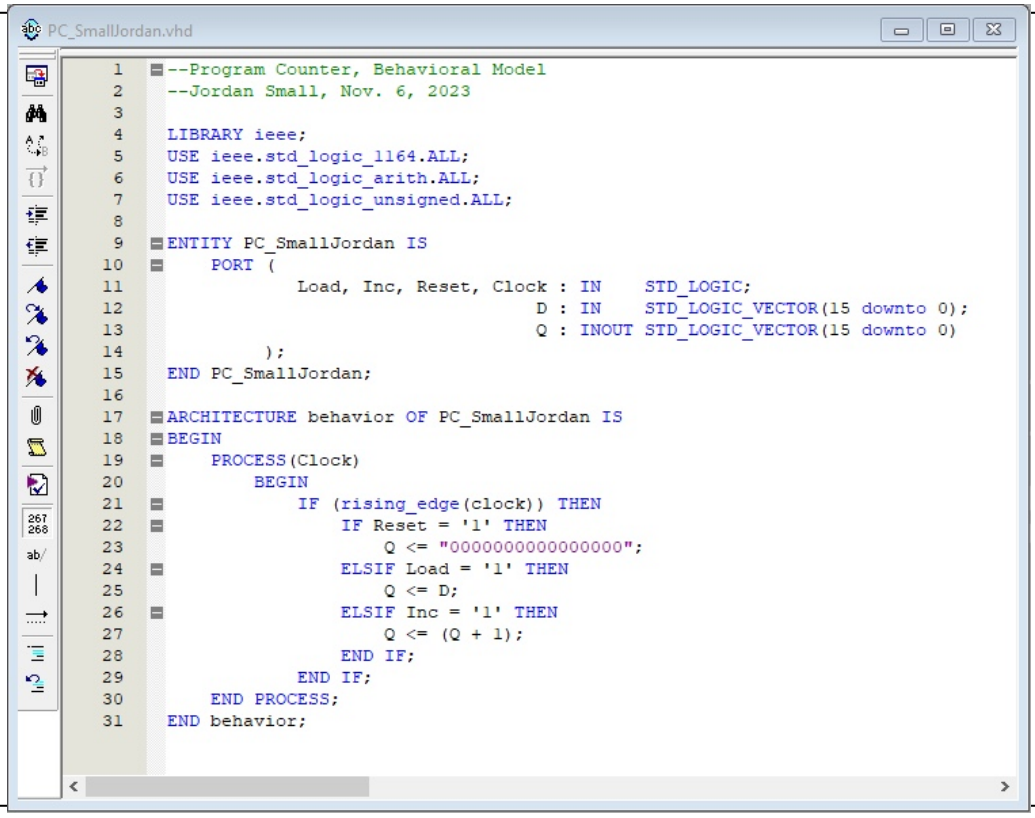


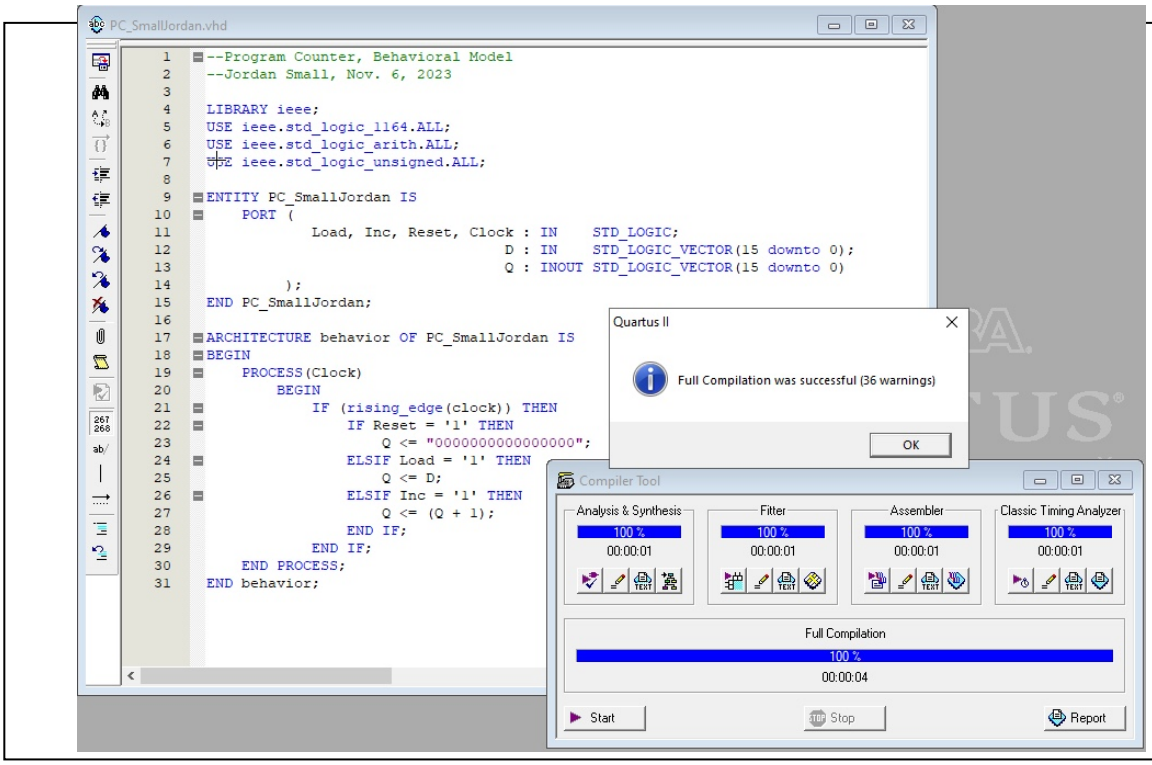

SixteenBitRegister_SmallJordan.vhd
--16-Bit Register, Behavioral Model
--Jordan Small, Nov. 6, 2023
LIBRARY ieee;
USE IEEE.std_logic_1164.ALL;
ENTITY SixteenBitRegister_SmallJordan IS
PORT (
Clock, Load : IN STD_LOGIC;
D : IN STD_LOGIC_VECTOR(15 downto 0);
Q : INOUT STD_LOGIC_VECTOR(15 downto 0)
);
END SixteenBitRegister_SmallJordan;
ARCHITECTURE behavior OF SixteenBitRegister_SmallJordan IS
BEGIN
PROCESS (Clock, Load)
BEGIN
IF(rising_edge(clock)) THEN
IF Load = '1' THEN
Q <= D;
END IF;
END IF;
END PROCESS;
END behavior;
Quartus II
Full Compilation was successful (36 warnings)
OK
Compiler Tool
Analysis & Synthesis
Fitter
Assembler
Classic Timing Analyzer
100 %
00:00:01
00:00:01
00:00:01
00:00:00
Full Compilation
100 %
00:00:03
Start
Stop
Report

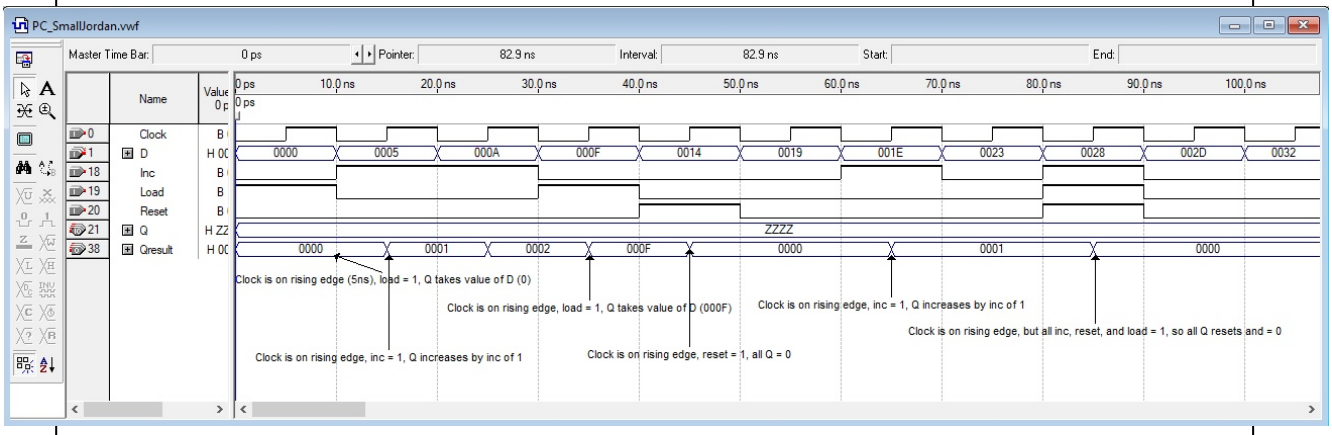

SixteenBitRegister_SmallJordan.vwf
Master Time Bar:
0 ps
Pointer:
30.15 ns
Interval:
30.15 ns
Start:
End:
Name
Clock
Load
Q
D
Qresult
0 ps
10.0 ns
20.0 ns
30.0 ns
40.0 ns
ZZZZZZZZZZZZZZZZ
0001
0002
0003
0004
0005
0000
0003
0004
Clock is on rising edge, load =1-- therefore Q[0,1,2] take values of D[0,1,2]
Clock is on rising edge, load = 1, therefore Q[0,1,2,3] take values of D[0,1,2,3]

2.4 – Program Counter Register

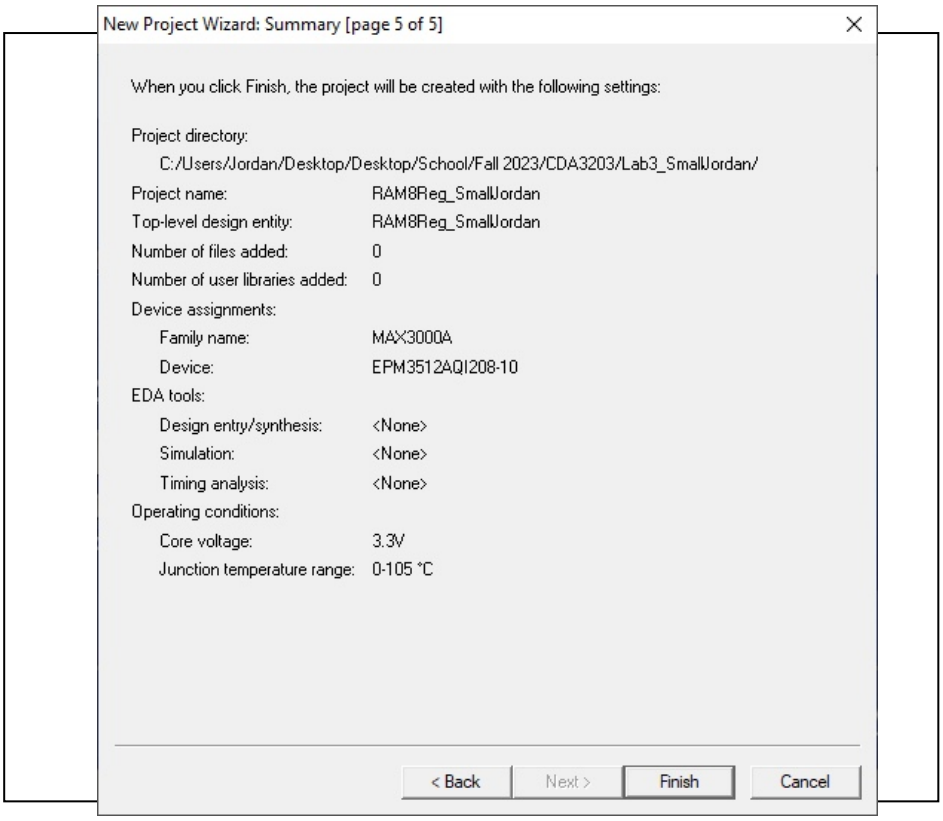


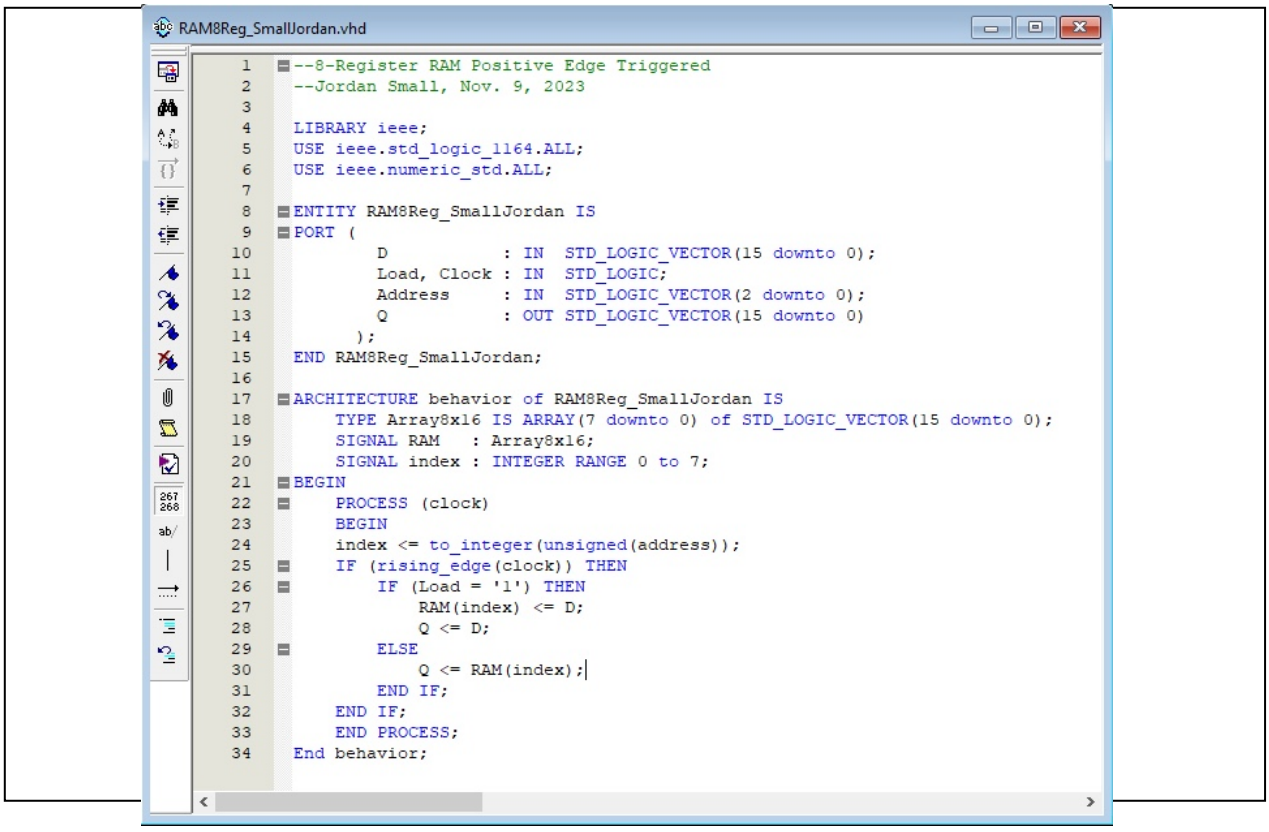


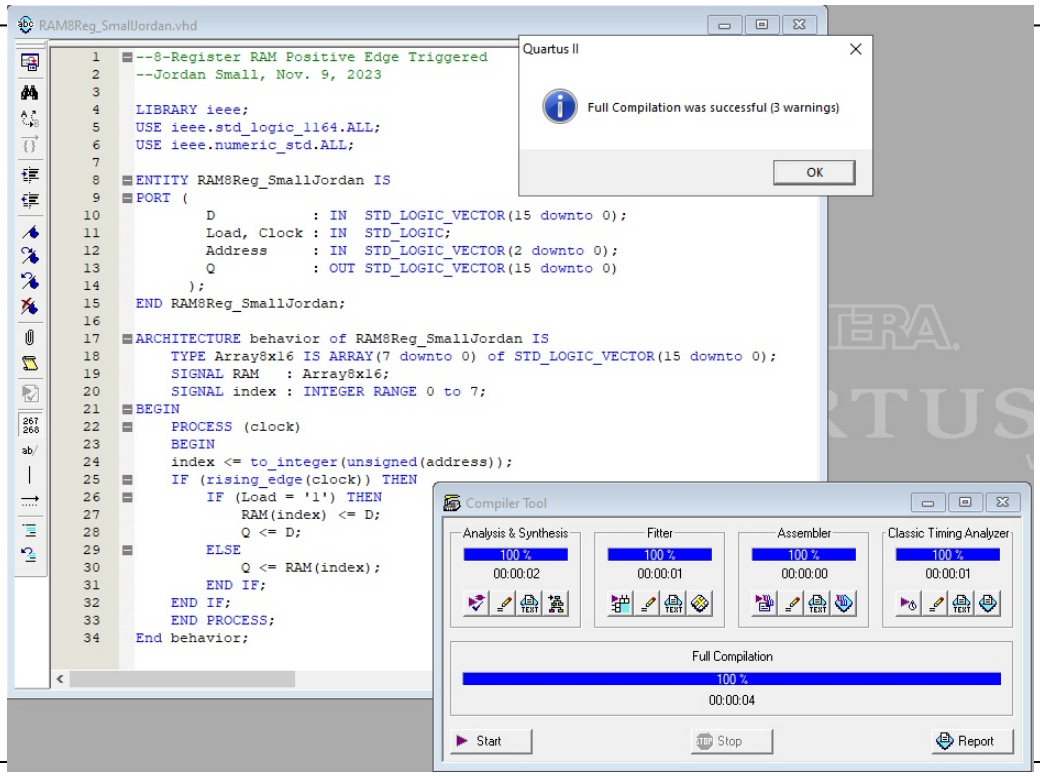

PC_SmallJordan.vhd
--Program Counter, Behavioral Model
--Jordan Small, Nov. 6, 2023
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_unsigned.ALL;
ENTITY PC_SmallJordan IS
PORT (
Load, Inc, Reset, Clock : IN STD_LOGIC;
D : IN STD_LOGIC_VECTOR(15 downto 0);
Q : INOUT STD_LOGIC_VECTOR(15 downto 0)
);
END PC_SmallJordan;
ARCHITECTURE behavior OF PC_SmallJordan IS
BEGIN
PROCESS(Clock)
BEGIN
IF (rising_edge(clock)) THEN
IF Reset = '1' THEN
Q <= "0000000000000000";
ELSIF Load = '1' THEN
Q <= D;
ELSIF Inc = '1' THEN
Q <= (Q + 1);
END IF;
END IF;
END PROCESS;
END behavior;
Quartus II
Full Compilation was successful (36 warnings)
OK
Compiler Tool
Analysis & Synthesis
Fitter
Assembler
Classic Timing Analyzer
100 %
00:00:01
Full Compilation
00:00:04
Start
Stop
Report

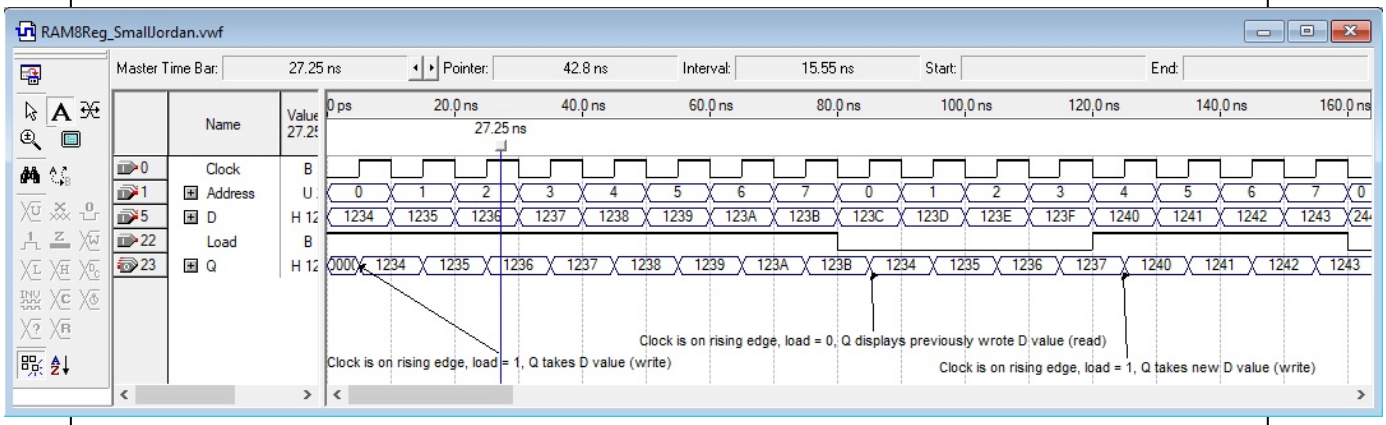

PC_SmallJordan.vwf
Master Time Bar:
Pointer:
Interval:
Start:
End:
Name
Clock
D
Inc
Load
Reset
Q
Qresult
0000
0005
000A
000F
0014
0019
001E
0023
0028
002D
0032
ZZZZ
0001
0002
Clock is on rising edge (5ns), load = 1, Q takes value of D (0)
Clock is on rising edge, load = 1, Q takes value of D (000F)
Clock is on rising edge, inc = 1, Q increases by inc of 1
Clock is on rising edge, reset = 1, all Q = 0
Clock is on rising edge, but all inc, reset, and load = 1, so all Q resets and = 0

2.5 – 8 Register RAM






RAM8Reg_SmallJordan.vhd
--8-Register RAM Positive Edge Triggered
--Jordan Small, Nov. 9, 2023
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
ENTITY RAM8Reg_SmallJordan IS
PORT (
D : IN STD_LOGIC_VECTOR(15 downto 0);
Load, Clock : IN STD_LOGIC;
Address : IN STD_LOGIC_VECTOR(2 downto 0);
Q : OUT STD_LOGIC_VECTOR(15 downto 0)
);
END RAM8Reg_SmallJordan;
ARCHITECTURE behavior of RAM8Reg_SmallJordan IS
TYPE Array8x16 IS ARRAY(7 downto 0) of STD_LOGIC_VECTOR(15 downto 0);
SIGNAL RAM : Array8x16;
SIGNAL index : INTEGER RANGE 0 to 7;
BEGIN
PROCESS (clock)
BEGIN
index <= to_integer(unsigned(address));
IF (rising_edge(clock)) THEN
IF (Load = '1') THEN
RAM(index) <= D;
Q <= D;
ELSE
Q <= RAM(index);
END IF;
END IF;
END PROCESS;
End behavior;
Quartus II
Full Compilation was successful (3 warnings)
OK
Compiler Tool
Analysis & Synthesis
100 %
00:00:02
Fitter
100 %
00:00:01
Assembler
100 %
00:00:00
Classic Timing Analyzer
100 %
00:00:01
Full Compilation
100 %
00:00:04
Start
Stop
Report


RAM8Reg_SmallJordan.vwf
Master Time Bar: 27.25 ns
Pointer: 42.8 ns
Interval: 15.55 ns
Start:
End:
Name
Value
Clock
Address
D
Load
Q
0 ps
20.0 ns
40.0 ns
60.0 ns
80.0 ns
100.0 ns
120.0 ns
140.0 ns
160.0 ns
27.25 ns
Clock is on rising edge, load = 1, Q takes D value (write)
Clock is on rising edge, load = 0, Q displays previously wrote D value (read)
Clock is on rising edge, load = 1, Q takes new D value (write)